ADOBE® AIR® 安全性

法律声明

目录

AIR 安全性概述

安全性是 Adobe、用户、系统管理员和应用程序开发人员关注的焦点。因此， Adobe® AIR® 包含一组安全性规则和控制，以保护用户和应用程序开发人员的利益。本白皮书介绍了在使用和开发 Adobe AIR 的应用程序时的一些安全注意事项。

虽然 AIR 安全模型是由用于 Flash® Player 中运行的 SWF 内容和浏览器中运行的 HTML 内容的安全模型发展而来的，但此安全协议与适用于浏览器中内容的安全协议不同。此协定为开发人员提供了一种自由访问更广泛的功能以便获得丰富体验的安全方式，而这种方式并不适合基于浏览器的应用程序。

AIR 应用程序按照与给定计算设备上其他本机应用程序相同的操作系统安全性约束运行。通常，使用这些约束可以对操作系统功能进行广泛的访问，例如读取和写入文件、绘制到屏幕以及与网络进行通信。适用于本机应用程序的操作系统限制 （例如特定于用户的权限）同样适用于 AIR 应用程序。

AIR 应用程序是采用编译过的字节代码 （SWF 内容）或解释过的脚本 （JavaScript、 HTML）编写的，以便运行时提供内存管理。这样可以最大程度地减少与内存管理 （如缓冲区溢出和内存损坏）有关的漏洞对 AIR 应用程序产生影响的可能性。下面是一些影响用本机代码编写的桌面应用程序的最常见漏洞。

注：本白皮书将讨论 Adobe AIR 中与安全相关的问题。下面的开发人员文档提供了有关开发安全的 AIR 应用程序的技术细节和使用 AIR API 的注意事项：

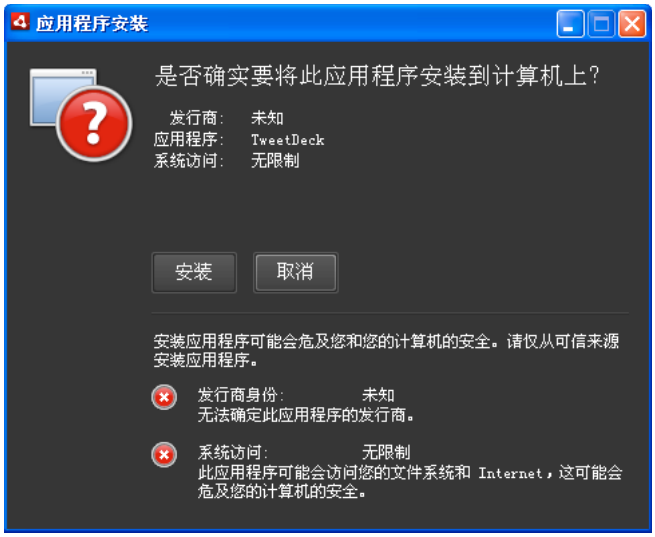
- 对于 ActionScript （Flash 和 Flex）开发人员，请参阅 《ActionScript 3.0 开发人员指南》中的 AIR 安全性
- 对于 Ajax 开发人员，请参阅 《Adobe AIR HTML 开发人员指南》中的 AIR 安全性。

安装和更新桌面应用程序

桌面 AIR 应用程序可以通过使用 air 扩展名的 AIR 安装程序文件分发。如果安装了 Adobe AIR 且打开了 AIR 文件，运行时将管理应用程序的安装过程。

注：开发人员可以指定版本、应用程序名称和发行商源，但初始应用程序安装流程本身无法修改。此限制对用户非常有利，因为所有 AIR 应用程序共享由 Adobe AIR 管理的安全、简单且一致的安装过程。如果有必要对应用程序进行自定义，则可以在首次执行应用程序时进行自定义。

默认的应用程序安装程序为用户提供了与安全相关的信息。用可信的证书或链 至安装计算机上可信证书的证书对 AIR 应用程序进行签名后， AIR 会在安装期间显示发行商名称。否则，发行商名称将显示为“未知”。因此，用户可以就是否安装应用程序做出明智的决定：



AIR 应用程序首先要求在用户的计算机上安装运行时，就像 SWF 文件首先要求安装 Flash Player 浏览器插件一样。

可以通过两种方式安装运行时：使用无缝安装功能或通过手动安装。

- 借助无缝安装功能，开发人员可以让没有 Adobe AIR 安装经验的用户体验简单化的安装过程。在无缝安装方法中，开发人员可以将 SWF 文件嵌入到网页中，该 SWF 文件会提供安装的 AIR 应用程序的名称。用户单击 SWF 文件安装应用程序时， SWF 文件会检查是否存在运行时。如果检测不到运行时，运行时会自行安装并且会立即激活，同时开始安装开发人员的应用程序。用户可以选择取消安装。
- 用户也可以在安装 AIR 文件之前手动下载并安装运行时。开发人员随后可以通过不同的方式 （例如电子邮件或网站上的 HTML 链接）分发 AIR 文件。打开 AIR 文件后，运行时即被激活并开始处理应用程序的安装。

AIR 安全模型允许用户决定是否要安装 AIR 应用程序。 AIR 安装程序对本机应用程序安装技术提供了若干改进，使得用户更方便地做出以下信任决策：

- 即使通过 Web 浏览器中的链接安装 AIR 应用程序，运行时也会对所有操作系统提供一致的安装体验。大多数本机应用程序安装体验根据浏览器或其他应用程序提供安全信息 （如果提供了安全信息）。
- AIR 应用程序安装程序可以识别应用程序的数据源 （反之，如果无法验证数据源，安装程序会予以明确说明），并在用户允许安装继续时提供应用程序可用权限的相关信息。
- 运行时会管理 AIR 应用程序的安装过程。 AIR 应用程序无法控制运行时使用的安装过程。

通常，用户不应安装来自其不信任源或无法验证源的任何应用程序 （包括 AIR 应用程序）。与其他可安装应用程序一样，对本机应用程序执行的安全验证也适用于 AIR 应用程序。

AIR 2 添加了对扩展桌面 AIR 应用程序的支持。这些应用程序使用本机安装程序文件进行安装：

- Mac OS 上的 DMG 文件
- Windows 上的 EXE 文件
- Linux 上的 RPM 或 DEB 数据包

创建本机安装程序后，您将无法再享有标准 AIR 安装和更新模型的优势。您将对安装体验负责，就像对于使用其他技术创建的本机应用程序一样。

AIR 3 将添加对捕获运行时捆绑的支持。在此部署模型中，应用程序不再使用用户计算机上安装的共享运行时。但是，您的应用程序包含其自己的私有 AIR 运行时副本。在此模型中，您对安装和用户更新体验负责。此外，由于 Adobe 将不会更新您的应用程序所使用的 AIR 运行时，当有适用的运行时安全修补发布时，您还应当负责更新您的应用程序。

更新 AIR 应用程序

开发和部署软件更新是本机代码应用程序面临的最大安全挑战之一。安装的 AIR 应用程序可以检查更新 AIR 文件的远程位置。如果存在适当的更新，则会下载并安装 AIR 文件，然后重新启动该应用程序。有关使用此方法提供新功能和响应潜在安全漏洞的详细信息，请参阅开发人员文档。

AIR 2 添加了对扩展桌面 AIR 应用程序的支持。这些应用程序使用本机安装程序文件进行安装，也使用这些文件进行更新：

- Mac OS 上的 DMG 文件
- Windows 上的 EXE 文件
- Linux 上的 RPM 或 DEB 数据包

内置 AIR Updater 类和 AIR 更新框架不支持更新通过本机安装程序安装的 AIR 应用程序。（有开放源项目可支持此类更新。）

更多帮助主题

RIASpace：Native Application Updater 项目

删除 AIR 应用程序

删除 AIR 应用程序的同时也将删除应用程序目录中的所有文件。但不会删除应用程序可能已写入应用程序目录外部的文件。删除 AIR 应用程序不会撤消 AIR 应用程序对应用程序目录外部的文件所做的更改。

安装和更新移动应用程序

移动 AIR 应用程序可以作为受支持平台的本机包进行分发。在 Android 上，软件包格式是 APK 文件；在 iOS 上，软件包格式是 IPA 文件。用户可以通过平台所支持的标准方式来下载和安装移动 AIR 应用程序。例如，通过 Android 上的 Market 和 iOS 上的 App Store。 AIR 应用程序的安装受到与平台上其他应用程序同样的限制。

在 Android 上， AIR 运行时是单独安装的，只要启动 AIR for Android 应用程序，就会将它激活。

在 iOS 设备 （如 iPhone）上， AIR 运行时不是单独安装的； iOS 上的每个 AIR 应用程序都是自包含应用程序。

通常，用户不应安装来自其不信任源或无法验证源的任何应用程序 （包括 AIR 应用程序）。与其他可安装应用程序一样，对本机应用程序执行的安全验证也适用于 AIR 应用程序。

AIR 3 添加了对 Android 上的捕获运行时捆绑的支持。在此部署模型中，您的应用程序不再使用安装在用户设备上的共享运行时。但是，您的应用程序包含其自己的私有 AIR 运行时副本。在此模型中，您对安装和用户更新体验负责。此外，由于 Adobe 将不会更新您的应用程序所使用的 AIR 运行时，当有适用的运行时安全修补发布时，您还应当负责更新您的应用程序。请注意， iOS 上所使用的部署模型一直使用捕获运行时。

更新移动 AIR 应用程序

开发和部署软件更新是本机代码应用程序面临的最大安全挑战之一。移动设备上的 AIR 应用程序可以使用本机平台更新机制。在 Android 上，该机制是 Android Market。在 iOS 上，该机制是 Apple iTunes App Store。

Adobe AIR 更新

Adobe 会定期使用新功能和修补更新 Adobe AIR。

桌面 **AIR** 更新
在桌面操作系统上，使用自动通知和更新功能， Adobe 可以在提供了 Adobe AIR 的更新版本时自动通知用户。

对 Adobe AIR 的更新可确保 Adobe AIR 正常工作，还可能包含对安全性的重要更改。 Adobe 建议用户，只要提供了新版本的 Adobe AIR，就更新到这个最新版本 （特别是在涉及安全性更新时）。

默认情况下，当启动 AIR 应用程序时，运行时检查更新是否可用。至少每两周执行一次更新检查。如果更新可用， AIR 在后台下载更新。

通过使用 AIR SettingsManager 应用程序，用户可以禁用自动更新功能。 http://airdownload.adobe.com/air/applications/SettingsManager/SettingsManager.air 上提供了可供下载的 AIR SettingsManager 应用程序。

Adobe AIR 的正常安装过程包括连接到 http://airinstall.adobe.com 以发送有关安装环境的基本信息，例如操作系统版本和语言。仅在每次安装完成后发出此信息且 Adobe 可通过此信息确认安装是否成功。未收集或传输任何个人可识别的信息。

移动 **AIR** 更新
在 Android 上， AIR 运行时的更新通过 Android Market 分发。

在 iOS 上，运行时与每个应用程序捆绑在一起。应用程序开发人员可以使用更新的 SDK 来重建其应用程序，以便并入错误修补程序、安全性更新和新的运行时功能。

代码签名

Adobe AIR 要求所有 AIR 应用程序都需要进行数字签名。代码签名是一种数字签名代码过程，用于确保软件和发布者身份的完整性。开发人员可以通过证书颁发机构 (CA) 颁发的证书或构建自签名证书对 AIR 应用程序进行签名。

使用公认的证书颁发机构 (CA) 颁发的证书对 AIR 文件进行数字签名，可以向用户提供重要保证：他们正在安装的应用程序没有被意外或恶意更改。使用公认的证书颁发机构 (CA) 颁发的证书对 AIR 文件进行数字签名还可以将开发人员标识为签名者（发布者）。AIR 可识别由 Verisign 和 Thawte 证书颁发机构颁发的代码签名证书。如果开发人员已使用 Verisign 或 Thawte 证书对 AIR 文件进行签名，则 AIR 应用程序安装程序在安装过程中将显示发布者名称。

如果 AIR 应用程序已使用可信证书签名，或者所使用的签名证书链接到安装计算机上的可信证书，则 AIR 应用程序安装程序在安装过程中将显示发布者名称。证书颁发机构 (CA) 在颁发高可靠性证书之前，将使用已建立的验证过程对发布者或开发人员的身份进行验证。

开发人员还可以使用他们自己创建的自签名证书对 AIR 应用程序进行签名。但是， AIR 应用程序安装程序会将这些应用程序表示为来自未经验证的发布者。

对 AIR 文件进行签名后，安装文件中将包含一个数字签名。此签名包括程序包的摘要，用于证实 AIR 文件自签名以来未经修改；此签名还包括有关签名证书的信息，用于证实发行商身份。

AIR 使用的公钥基础结构 (PKI) 通过操作系统的证书存储支持。安装 AIR 应用程序的计算机必须直接信任用于对 AIR 应用程序签名的证书，或者该计算机必须信任将证书链接到可信证书颁发机构的证书链，以便验证发布者信息。

如果 AIR 文件用未链至其中一个受信根证书 （通常，这些证书包括所有自签名证书）的证书进行签名，则无法核实发行商信息。尽管 AIR 可以确定该 AIR 文件自签名后没有被更改，但无法验证实际创建和签署该文件的人员。

有关代码签名过程和认可的证书格式的详细信息，请参阅开发人员文档。

使用桌面本机安装程序的代码签名

将应用程序打包成本机安装程序时，可以选择应用本机代码签名。只有 Windows 支持本机代码签名。请参阅 MSDN: Introduction to Code Signing。

移动平台上的代码签名

在移动平台上，根据平台约定和要求对 AIR 应用程序进行签名。开发人员使用 AIR SDK 中的工具和满足移动平台要求的证书对其应用程序进行签名。移动 AIR 应用程序的安装由设备操作系统处理，而不是由 AIR 运行时处理。因此， AIR 不会对应用程序签名或证书持有者身份进行验证。

更多帮助主题

对 AIR 应用程序进行签名

安全沙箱

AIR 提供了一个全面的安全体系结构，用于定义 AIR 应用程序中每个文件的权限。这包括随应用程序一起安装的那些文件和应用程序加载的其他文件。根据文件的来源授权其相应的权限，并将其分配到称为沙箱的逻辑安全组中。

随应用程序一起安装的文件在称为 “ 应用程序目录 ” 的目录中，因此，在默认情况下，这些文件将放在称为 “ 应用程序沙箱 ” 的安全沙箱中，该沙箱拥有访问所有 AIR API 的权限。如果其中某些 API 可供除应用程序资源目录之外的源中的内容 （即那些没有随应用程序一起安装的文件）使用，则会带来巨大安全风险。

沙箱的 AIR 安全模型由 Flash Player 安全模型以及应用程序沙箱组成。不在应用程序沙箱中的文件具有类似 Flash Player 安全模型指定的安全限制。

运行时使用这些安全沙箱定义文件可以访问的数据范围以及可以执行的操作。若要维护本地安全，请将各个沙箱中的文件进行隔离。例如，从外部 Internet URL 加载到 AIR 应用程序的 SWF 文件放置在远程沙箱中，该文件默认情况下不具有通过脚本访问应用程序目录中分配给应用程序沙箱的文件的权限。

注：在 iOS 上，不允许执行下载的代码。

应用程序沙箱中的内容的权限

安装应用程序时， AIR 安装程序文件中包括的所有文件都会安装到用户计算机的应用程序目录中。在应用程序运行时，应用程序目录树中的所有文件都会分配到应用程序沙箱中。应用程序沙箱中的内容具有 AIR 应用程序的完全访问权限，包括与本地文件系统内容进行交互。

许多 AIR 应用程序只能使用这些本地安装的文件来运行应用程序。但是，不会限制 AIR 应用程序仅加载应用程序目录中的文件，它们可以加载任意源中任何类型的文件。这包括用户计算机上的文件以及外部源中的文件 （例如本地网络或 Internet 上的文件）。文件类型不会对安全限制产生影响；加载的 HTML 文件与从相同源加载的 SWF 文件具有相同的安全权限。（但是，应用程序沙箱中的内容不能加载沙箱外的 JavaScript 文件。详细信息请参阅开发人员文档。）

应用程序安全沙箱中的内容可以访问 AIR API，而其他沙箱中的内容则无法访问。例如，只有应用程序安全沙箱中的内容才可以读取和写入本地文件系统。

一些 JavaScript 技术可将字符串动态转换为可执行代码。从应用程序 URL 加载代码时，应用程序安全沙箱中的内容只能使用这些技术。在应用程序沙箱中使用这些技术会带来安全风险。例如，应用程序可能会在无意中执行从网络沙箱加载的字符串，而该字符串可能包含恶意代码，如删除或更改用户计算机上的文件或者将本地文件内容报告回不受信任的网络域的代码。详细信息请参阅开发人员文档。

注：在移动 AIR 应用程序中，无法将 HTML 和 JavaScript 加载到应用程序沙箱中。移动 AIR 应用程序使用系统 Web 控件显示这种内容。该控件具有和默认系统 Web 浏览器相同的安全注意事项。

非应用程序沙箱中的内容的权限

从网络或 Internet 位置加载的文件会分配到非应用程序沙箱。这些内容在行为方式上具有一组与 Web 浏览器中运行的 SWF 内容（在 Flash Player 中）或 Web 浏览器中运行的 HTML 内容相同的权限和限制。

与应用程序安全沙箱中的内容不同，非应用程序安全沙箱中的 HTML 代码在任何时候都可以使用 JavaScript 方法执行动态生成的代码。

非应用程序沙箱中的代码无权访问提供应用程序功能的受权限保护的 AIR API。

详细信息请参阅开发人员文档。

访问文件系统

在 Web 浏览器中运行的应用程序只能与用户的本地文件系统进行有限的交互。 Web 浏览器会实施安全策略，用于确保用户的计算机不会由于加载 Web 内容而被破坏。例如，通过 Flash Player 在浏览器中运行的 SWF 文件无法直接与用户计算机中的文件进行交互。可以将共享对象写入用户的计算机，以便维护用户首选参数和其他数据，但文件系统交互将受到此限制。由于 AIR 应用程序安装在本地，因此它们与最终用户之间有不同的安全协议。应用程序与最终用户之间的这一协议是在安装时建立的，就像本机应用程序一样，其中包括应用程序在本地文件系统中读取和写入的功能。

这一灵活性要求开发人员担负较高的责任。意外的应用程序安全漏洞不仅会危害应用程序的功能，而且会危害用户计算机的完整性。开发人员文档中包括介绍最佳做法的 AIR 安全性信息。

除非用户计算机应用了管理员限制，否则 AIR 应用程序有权写入用户硬盘上的任意位置。但是，建议开发人员使用运行时为每个应用程序提供的特定于用户和特定于应用程序的应用程序存储目录。利用 AIR API，开发人员可以方便地读取应用程序存储目录中的数据和向其中写入数据。该运行时还为每个应用程序和用户提供唯一的加密本地数据存储区域。这允许应用程序以加密格式保存和检索存储在用户本地硬盘上的数据，其他应用程序或用户无法解密这些数据。为每个 AIR 应用程序使用一个单独的加密本地存储区，每个 AIR 应用程序为每个用户使用一个单独的加密本地存储区。应用程序可以使用加密的本地存储来存储必须保护的信息，如用于 Web 服务的登录凭据。AIR 使用 DPAPI（在 Windows 上）和 KeyChain（在 Mac OS 上）将加密的本地存储与每个用户相关联。加密的本地存储区使用 AES-CBC 128 位加密。

在 Adobe AIR 2 中，应用程序可以使用为文件类型注册的默认应用程序打开文件。例如，应用程序可以使用用于打开 mp3 文件的默认应用程序打开 mp3 文件。AIR 会阻止应用程序打开包含特定文件类型的文件。这些文件类型在打开时可能执行代码。Windows 中的 EXE 文件就是这种文件。用于 Adobe Flash Platform 的 ActionScript 3.0 参考 中列出了受限制的文件类型。但是，使用本机安装程序安装的扩展桌面 AIR 应用程序可以打开任何类型的文件。（有关扩展桌面应用程序的信息，请参阅第 6 页的 “ 与本机进程通信 ”。）

与本机进程通信

自 Adobe AIR 2 起，桌面 AIR 应用程序可以通过命令行运行并与其他本机进程通信。例如， AIR 应用程序可以运行进程，并通过标准的输入和输出流与之通信。

要与本机进程通信，开发人员需要通过本机安装程序将要安装的 AIR 应用程序打包。本机安装程序的文件类型取决于目标操作系统：

- 在 Mac OS 中为 DMG 文件。
- 在 Windows 中为 EXE 文件。
- 在 Linux 中为 RPM 或 DEB 包。

这些应用程序称为扩展的桌面配置文件应用程序。

打包这些应用程序时，开发人员要使用代码签名证书对应用程序进行签名。所用证书与对标准桌面 AIR 应用程序进行签名时使用的证书种类相同。

本机进程 API 可以在用户系统上运行任何可执行文件。 AIR 文档为开发人员提供了有关安全使用本机进程 API 的指导。该文档提醒开发人员在构造和执行命令时要谨慎。应用程序应该对发送给本机进程的数据进行验证。

Windows 中的 AIR 会阻止扩展桌面应用程序直接运行 .bat 文件。.bat 文件的命令行参数可能包含额外字符的潜在恶意注入。这些注入可能导致 cmd.exe 应用程序 （该程序执行 .bat 文件）执行有害或不安全的应用程序。

安全使用不受信任的内容

未分配给应用程序沙箱的内容可以为 AIR 应用程序提供其他脚本功能，但前提是满足运行时的安全条件。本节介绍 AIR 安全协议以及非应用程序内容。

AIR 应用程序限制脚本访问非应用程序内容比 Flash Player 浏览器插件限制脚本访问不受信任内容更严格。例如，在浏览器中的 Flash Player 中， SWF 文件可以调用 System.allowDomain() 方法，对从指定域加载的任何 SWF 内容授予脚本访问权限。AIR 应用程序安全沙箱中的内容不允许调用此方法，以防止向非应用程序文件授予对用户文件系统的不合理访问权限。

通过脚本访问应用程序和非应用程序内容的 AIR 应用程序具有更复杂的安全安排。只允许应用程序沙箱外的文件使用沙箱桥来访问应用程序沙箱中的文件的属性和方法。沙箱桥充当应用程序内容与非应用程序内容之间的通道，在两个文件之间提供显式交互。如果使用正确，沙箱桥会提供额外的安全层，从而限制非应用程序内容访问属于应用程序内容的对象引用。

通过示例可以更好地说明沙箱桥的优点。假设 AIR 音乐商店应用程序需要为希望创建自己的 SWF 文件的广告商提供 API，商店应用程序可以使用这些文件进行通信。该商店需要为广告商提供在商店中查找艺术家和光盘的方法，另外出于安全原因，还需要将某些方法和属性与第三方 SWF 文件进行隔离。

沙箱桥可以提供此功能。默认情况下，在运行时从外部加载到 AIR 应用程序的内容无法访问主应用程序中的任何方法或属性。通过自定义沙箱桥，开发人员可以在不公开这些方法或属性的情况下为远程内容提供服务。沙箱桥在受信任和不受信任内容之间架起了一个有限通道。

有关使用沙箱网桥的完整详细信息，请参阅：

- 对于 ActionScript （Flash 和 Flex）开发人员，请参阅 《ActionScript 3.0 开发人员指南》中的 AIR 安全性。
- 对于 Ajax 开发人员，请参阅 《Adobe AIR HTML 开发人员指南》中的 AIR 安全性。

Android 设备上的安全性

在 Android 上，和在所有计算设备上一样， AIR 符合本机安全模型。同时， AIR 保持自己的安全性规则，旨在使开发人员轻松编写安全的、与 Internet 连接的应用程序。

因为 Android 上的 AIR 应用程序使用 Android 软件包格式，安装属于 Android 安全性模型。不使用 AIR 应用程序安装程序。

Android 安全性模型有三个主要方面：

- 权限
- 应用程序签名
- 应用程序用户 ID

Android 权限

Android 的许多功能由操作系统权限机制保护。为了使用某种保护的功能， AIR 应用程序描述符必须声明应用程序要求必要的权限。用户尝试安装应用程序时， Android 操作系统会在继续安装之前对用户显示所有请求的权限。

大多数 AIR 应用程序需要在应用程序描述符中指定 Android 权限。默认情况下，不包括任何权限。通过 AIR 运行时公开的受保护的 Android 功能需要下列权限：

ACCESS_COARSE_LOCATION 允许应用程序通过 Geolocation 类访问 WIFI 和移动电话网络位置数据。

ACCESS_FINE_LOCATION 允许应用程序通过 Geolocation 类访问 GPS 数据。

ACCESS_NETWORK_STATE 和 ACCESS_WIFI_STATE 允许应用程序通过 NetworkInfo 类访问网络信息。

CAMERA 允许应用程序访问摄像头。

INTERNET 允许应用程序提出网络请求。也允许远程调试。

READ_PHONE_STATE 允许 AIR 运行时在有来电时静音。

RECORD_AUDIO 允许应用程序访问麦克风。

WAKE_LOCK 和 DISABLE_KEYGUARD 允许应用程序使用 SystemIdleMode 类设置阻止设备休眠。

WRITE_EXTERNAL_STORAGE 允许应用程序写入设备上的外部存储卡。

应用程序签名

为 Android 平台创建的所有应用程序包都必须进行签名。由于 Android 上的 AIR 应用程序都以本机 Android APK 格式打包，因此根据 Android 约定而非 AIR 约定对其进行签名。尽管 Android 和 AIR 使用代码签名的方式相似，但存在显著区别：

- 在 Android 上，签名将验证开发人员是否持有私钥，但不用于验证开发人员的身份。
- 对于提交给 Android 市场的应用程序，证书必须至少在 25 年内有效。
- Android 不支持将包签名迁移到其他证书。如果更新通过其他证书进行签名，则用户必须卸载原始应用程序后才能安装更新的应用程序。
- 两个使用相同证书签名的应用程序可以指定一个共享的 ID，以便允许它们访问彼此的缓存和数据文件。（这种共享不是由 AIR 促成的。）

应用程序用户 **ID**

Android 使用 Linux 内核。为每个安装的应用程序分配了 Linux 类型的用户 ID，该用户 ID 确定其进行文件访问等操作的权限。通过文件系统权限提供保护，防止对应用程序、应用程序存储和临时目录中的文件进行非授权访问。写入外部存储器 （即 SD 卡）的文件在 SD 卡作为大容量存储设备安装到计算机上时，可以被其他应用程序或用户读取、修改和删除。

通过 Internet 请求接收的 Cookie 不会在各个 AIR 应用程序之间共享。

更多帮助主题

Android：安全性和权限

应用程序安装

默认情况下， Android 上的 AIR 应用程序使用一个由 Adobe 维护和更新的共享运行时库。从 AIR 3 开始，您可以将您的应用程序与一个 “ 捕获的 ” 运行时绑定。装有捕获的运行时的应用程序使用该版本的运行时 — 而不是共享的 AIR 运行时，该运行时也可以存在于设备上。当发布 AIR 运行时的新版本时，不会自动更新捕获的运行时。

重要说明：如果使用捕获的运行时，当 Adobe 发布相关的安全更新时，您应负责更新该运行时。

Android 上的加密数据

Android 上的 AIR 应用程序可以使用内置 SQL 数据库中提供的加密选项保存加密数据。

可以使用 EncryptedLocalStore 类保存数据，但不会加密该数据。相反， Android 安全模型依赖于应用程序用户 ID 来保护其他应用程序的数据。使用共享用户 ·ID 并使用相同代码签名证书签名的应用程序使用相同的加密本地存储。

重要说明：在根电话上，使用根权限运行的任何应用程序都可以访问任何其他应用程序的文件。因此，使用加密的本地存储所存储的数据在根设备上并不安全。

在 iOS 设备上的安全性

在 iOS 上， AIR 符合本机安全模型。同时， AIR 保持自己的安全性规则，旨在使开发人员轻松编写安全的、与 Internet 连接的应用程序。

因为 iOS 上的 AIR 应用程序使用 iOS 软件包格式，安装属于 iOS 安全性模型。不使用 AIR 应用程序安装程序。此外，在 iOS 设备上不使用单独的 AIR 运行时。所有 AIR 应用程序都包含运行所需的全部代码。

重要说明：由于运行时代码作为您的应用程序的一部分包括在内，因此在 Adobe 发布 AIR 的新版本时，不会自动更新该运行时代码。当 Adobe 发布相关的安全更新时，您应负责更新您的应用程序。

应用程序签名
必须对所有针对 iOS 平台创建的应用程序包进行签名。因为 iOS 上的 AIR 应用程序都打包为本机 iOS IPA 格式，所以它们是根据 iOS 要求 （而不是 AIR 要求）进行签名的。虽然 iOS 和 AIR 以类似的方式使用代码签名，但也存在明显的差异：

- 在 iOS 上，用于对应用程序进行签名的证书必须由 Apple 颁发；无法使用来自其他证书颁发机构的证书。
- 在 iOS 上， Apple 颁发的分发证书的有效期一般为一年。

HTML 安全性

HTML 内容的安全注意事项不同于基于 SWF 的内容，主要是因为 JavaScript 能够创建动态生成的代码。如果在应用程序沙箱内允许使用动态生成的代码 （如调用 eval() 函数时生成的代码），则会引起安全性风险。例如，应用程序可能会在无意中执行从网络沙箱加载的字符串，而该字符串可能包含恶意代码，如删除或更改用户计算机上的文件或者将本地文件内容报告回不受信任的网络域的代码。

生成动态代码的方式如下所示：

- 调用 eval() 函数。
- 设置 innerHTML 属性或调用 DOM 函数，以插入脚本标签，从而直接加载资源外部的脚本。
- 设置 innerHTML 属性或调用 DOM 函数以插入具有内联代码的脚本标签 （而非通过 src 加载脚本）。
- 将应用程序沙箱中内容的 script 标签的 src 设置为一个不在应用程序资源目录中的文件。
- 使用 javascript URL 方案 （如 href="javascript:alert('Test')" 所示）。

从应用程序目录加载内容时，应用程序安全沙箱中的代码只能使用这些方法。这可以防止应用程序沙箱 （具有对所有 AIR API 的访问权限）中的代码执行可能来自不受信任源中的脚本。

非应用程序安全沙箱中的内容可以使用这些方法生成动态代码。但是，它们不能直接访问 AIR API。AIR 沙箱桥功能为非应用程序安全沙箱中的代码提供了一种与应用程序沙箱中的代码进行交互的方式 （交互方法由应用程序代码限制和决定）。

AIR 应用程序可以从字符串变量生成 HTML 内容 （而不是从文件或网络源中加载）。但默认情况下，由字符串生成的 HTML 内容不会被授予应用程序沙箱权限。这样可防止应用程序在不经意间授予对从潜在不安全 Internet 源获得的字符串内容的访问权限。

注：在移动设备上，AIR 使用主机操作系统提供的 Web 控件。该控件中运行的内容不具有访问 AIR API 的权限，而且永远不会在应用程序安全沙箱中加载或执行。

有关 HTML 安全性的详细信息，请参阅开发人员文档中的“AIR 安全性”：

- 对于 ActionScript （Flash 和 Flex）开发人员，请参阅 《ActionScript 3.0 开发人员指南》中的 AIR 安全性。
- 对于 Ajax 开发人员，请参阅 《Adobe AIR HTML 开发人员指南》中的 AIR 安全性。

  另请参阅 Adobe AIR HTML 安全性白皮书。

其他安全注意事项

虽然 AIR 应用程序是使用 Web 技术构建的，但开发人员应知道这些应用程序并不在浏览器安全模型中运行，这一点很重要。这意味着，可以构建会对本地系统有意或无意产生损害的 AIR 应用程序。AIR 会尝试最大程度降低此风险，但仍存在一些可能引入漏洞的方式。本节介绍了重要的潜在不安全因素。有关构建避免这些风险的应用程序的最佳做法，请参阅开发人员文档。

将文件导入应用程序安全沙箱的风险

应用程序沙箱中的内容具有运行时的完全权限。建议开发人员考虑以下问题：

- 仅在必要时才在 AIR 文件 （位于安装的应用程序中）中包含文件。
- 仅在脚本文件的行为被完全理解和信任时才在 AIR 文件 （位于安装的应用程序中）中包含该脚本文件。
- 不要将网络源中的数据用作可能引起代码异常的 AIR API 的方法的参数。

  Adobe AIR 可防止应用程序沙箱中的内容将来自网络的数据作为代码使用，以免无意中执行恶意代码。这包括使用 ActionScript Loader.loadBytes() 方法和 JavaScript eval() 函数。

使用外部源确定路径的风险

使用外部数据或内容可能会破坏 AIR 应用程序。因此，应用程序在使用网络或文件系统中的数据时应特别小心。信任责任主要取决于开发人员以及他们建立的网络连接，但加载外来数据本身具有风险，不应将其用作敏感操作的输入。建议开发人员不要执行以下操作：

- 使用网络源中的数据确定文件名
- 使用网络源中的数据构建应用程序用来发送私人信息或启动其他应用程序的 URL

使用、存储或传输无保护凭据的风险

将用户凭据存储在用户的本地文件系统中将引入可能破坏这些凭据的风险。建议开发人员考虑以下问题：

- 如果凭据必须存储在本地，请在写入本地文件系统时对凭据进行加密。 Adobe AIR 为每个安装的应用程序都提供唯一的加密存储区，开发人员文档中对此进行了详细介绍。
- 不要将未加密的用户凭据传送到网络源，除非该来源受信任，并且传输协议使用传输层安全性 (TLS)，即使用 https: 或 SecureSocket 类。
- 永远不要在创建凭据时指定默认密码，应让用户自己创建密码。保留默认值的用户会将其凭据公开给已经知道默认密码的攻击者。

接收或传送远程数据的风险

当信息通过 Internet 进行传输时，有可能被监听和更改。您可以使用传输层安全 (TLS) 或更旧的安全套接字层 (SSL) 协议对服务器和客户端系统之间的通信进行加密。使用 HTTPS 协议保护超文本协议 (HTTP) 通信。使用 SecureSocket 类保护 TCP 套接字通信。在这两种情况下，提供数据的服务器必须使用 TLS 或 SSL，以便使您的 AIR 应用程序能安全地发送或接收数据。SecureSocket 类已在 AIR 2 中添加，可由 AIR 应用程序沙箱中运行的内容使用。建议开发人员考虑以下问题：

- 对服务器所承载的使用所支持的最新版本的 TLS 协议的敏感数据提供访问。（目前， AIR 支持 TLS 版本 1 和 SSL 版本 4。）
- 使用 HTTPS 传送或接收使用 HTTP 协议的敏感数据。
- 使用 SecureSocket 类传送或接收使用 TCP 套接字的敏感数据。

降级攻击的风险

在安装应用程序过程中，运行时会进行检查以确保应用程序的版本不是当前安装的版本。如果应用程序已经安装，则运行时会将现有应用程序的版本字符串与正在安装的版本进行比较。如果此字符串不同，则用户可以选择升级安装。运行时无法保证新安装的版本比旧版本新，仅保证版本不同。攻击者可能会向用户分发旧版本以避开安全漏洞。采取相应措施可以降低这种风险，开发人员文档提供了实现规避风险的版本架构和更新检查的最佳做法。